



#14/A
APPEAL
FEB 27 2004
USSN: 09/757,431
1851P/CA92000-0014

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

In Re Application of:

Date: February 23, 2004

LINDSAY, et al.

Confirmation No.: 8483

Serial No.: 09/757,431

Group Art Unit: 2172

Filed: January 10, 2001

Examiner: Ly, Anh

For: EFFICIENT INDEX-DATA FETCH VIA CALLBACK FOR TABLE DATA

Honorable Commissioner of Patents and Trademarks
Washington, D.C. 20231

RECEIVED

MAR 03 2004

Technology Center 2100

APPEAL BRIEF TRANSMITTAL LETTER

Sir:

Submitted herewith are an original and two copies of an Appellant's Brief on Appeal under 37 C.F.R. § 1.192 in connection with the above-referenced Patent application. The Brief includes an Appendix.

Check no. 5790 in the amount of \$330.00 is enclosed for payment of the Appeal Brief filing fee. A duplicate of this sheet is attached.

Respectfully submitted,


Joyce Tom
Attorney for Applicants
Reg. No. 48,681
(650) 493-4540



USSN: 09/757,431

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

APPEAL NO:

In Re Application of: LINDSAY, et al.

Serial No: 09/757,431

Filed: January 10, 2001

For: EFFICIENT INDEX-DATA FETCH VIA CALLBACK FOR TABLE DATA

RECEIVED

MAR 03 2004

Technology Center 2100

03/02/2004 HYUONG1 00000096 09757431

01 FC:1402

330.00 OP

APPELLANT'S BRIEF

Joyce Tom
Attorney for Appellants
International Business Machines
Sawyer Law Group, LLP
2465 E. Bayshore Road, Suite 406
Palo Alto, CA 94303



USSN: 09/757,431

**IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES**

In Re Application of:

Date: February 23, 2004

LINDSAY, et al.

Confirmation No.: 8483

Serial No.: 09/757,431

Group Art Unit: 2172

Filed: January 10, 2001

Examiner: Ly, Anh

For: EFFICIENT INDEX-DATA FETCH VIA CALLBACK FOR TABLE DATA

Honorable Commissioner of Patents and Trademarks
Washington, D.C. 20231

RECEIVED

MAR 03 2004

APPELLANT'S BRIEF ON APPEAL

Technology Center 2100

Sir:

Appellant herein files an Appeal Brief drafted in accordance with the provisions of 37 C.F.R. § 1.192(c) as follows:

I. REAL PARTY IN INTEREST

Appellants respectfully submit that the above-captioned application is assigned, in its entirety to International Business Machines Corporation, of Armonk, New York.

II. RELATED APPEALS AND INTERFERENCES

Appellants state that, upon information and belief, they are not aware of any co-pending appeal or interference which will directly affect or be directly affected by or have a bearing on the Board's decision in the pending appeal.

III. STATUS OF CLAIMS

Claims 1-14, 56 and 57 are pending in the present application and stand rejected. Claims 1, 2, 8, 9 and 12 were amended, and claims 56 and 57 added in a Response filed on April 28, 2003. Accordingly, Claims 1-14, 56 and 57 are on appeal and all applied rejections concerning those claims are herein being appealed.

IV. STATUS OF AMENDMENT

All amendments have been entered.

V. SUMMARY OF THE INVENTION

The present invention is directed to a method for processing a database query. In accordance with the present invention, a data manager utilizes an index manager to locate a data identifier (data ID) in an index corresponding to a selected key value. Instead of returning the located data ID to the data manager, the index manager issues a callback and passes the located data ID to the data manager. The data manager then determines whether the data specified by the data ID will be returned to a runtime (e.g., if it satisfies a query predicate and/or is not consumed). If the specified data is not returned, the index manager receives such an indication from the data manager, and subsequently does not return the data ID, but rather continues the index-data fetch for the next qualifying data ID. (Specification, page 9, line 20 to page 10, line 13).

The present invention, as recited in claim 1, provides:

1. A method for processing a database query on a set of data in a database management system having a data manager and an index manager, the method comprising the steps of:

- a) in response to a data manager call to locate a data identifier in an index corresponding to a selected key value, performing the steps of:
 - i) locating the data identifier in the index for the selected key value; and
 - ii) issuing a callback to the data manager; and
- b) continuing to carry out the an index-data fetch for another data identifier if there is another data identifier for the selected key value in the index and the index manager receives a specific condition from the data manager in response to the callback.

Independent claim 8 is a computer product claim having similar scope to that of claim 1.

VI. ISSUES

The issue presented is:

1. Whether claims 1-14, 56 and 57 are unpatentable under 35 U.S.C. § 103(a) over Ponnekanti (U.S. Patent No. 6,363,387) in view of Edwards et al (U.S. Patent No. 6,353,820).

VII. GROUPING OF CLAIMS

Appellant hereby states that claims 1, 11, 21, 31 and 37 do not stand or fall together, but rather claims 1-7 and 56 form one group, and claims 8-14 and 57 form another group.

VIII. ARGUMENTS

A. Summary of the Applied Rejections

In the Final Office Action, the Examiner rejected claims 1-14, 56 and 57 under 35 U.S.C. §103(a) as being unpatentable over Ponnekanti (U.S. Patent No. 6,363,387) in view of Edwards et al (U.S. Patent No. 6,353,820). In so doing, the Examiner stated:

With respect to claim 1, Ponnekanti discloses in response to a data

manager (data page table for query processing: col. 8, lines 6-20; also see fig. 2B); call to locate a data identifier in an index corresponding to a selected key value (data records or rows of database table: RID or page ID: col. 8, lines 18-20, and also see col. 4, lines 31-54), performing the step of locating the data identifier in the index for the selected key value (col. 16, lines 5-30); and continuing to carry out the index-data fetch for another data identifier (scanning for the next qualifying row: col. 16, lines 28-30 and lines 54-55), if there is another data identifier for the selected key value in the index, and the index manager receives a specific condition from the data manager (col. 16, lines 12-44).

As to the limitation, "issuing a callback," Ponnekanti does not explicitly indicate the callback.

However, Edwards discloses issuing the call to the calling program for index key value in the searching index key (col. 7, lines 10-18 and col. 5, lines 28-33).

B. The Cited Prior Art

Ponnekanti

Ponnekanti is directed to minimizing locking to optimize concurrency. Ponnekanti discloses deferred locking where the Index Manager returns RIDs to the Data Layer without acquiring locks on them during scans that have scan arguments on columns not in the index. The Data Layer subsequently qualifies the data row and determines whether locking is really necessary. Thus, the locking is done after the Data Layer reads the data row and determines that the data row qualifies.

Edwards

Edwards is directed to improving index processing performance in a multi-layer RDBMS by utilizing a subroutine designed to execute functions performed by lower component layers. Under certain conditions (see Figure 4b), the subroutine allows the RDBMS to bypass lower level component layers (RFM and I/O layers) during an index search, thereby reducing the time

in which such a search is performed.

B. Independent Claims 1 and 8 Are Allowable Over the Cited References.

Applicants respectfully submit that Ponnekanti in view of Edwards fails to teach or suggest the present invention as recited in claims 1 and 8. In particular, neither reference teaches or suggests “issuing a callback to the data manager” and “continuing to carry out an index-data fetch for another data identifier if . . . the index manager receives a specific condition from the data manager in response to the callback,” as recited in claims 1 and 8. According to the present invention, the index manager issues a callback to the data manager instead of returning the data ID to the data manager. The index manager then waits to receive a specific condition from the data manager before it continues the index-data fetch.

In contrast to the present invention, Ponnekanti’s Index Manager returns the RID to the Data Layer when it locates and qualifies the RID. (Col. 15, lines 30-56, col. 16, lines 6-7). Applicants agree with the Examiner that nothing in Ponnekanti teaches or suggests that the Index Manager issues “a callback to the data manager” when it locates a qualifying data ID, as recited in claims 1 and 8. The Examiner, however, contends that Edwards teaches this feature.

Applicants respectfully submit that Edwards fails to teach or suggest an index manager “issuing a callback to the data manager” after it has located a data ID for a selected key value, as recited in claims 1 and 8. According to Edwards, the output code component 204-6 (Figure 3a) calls the index enhancing performance subroutine IP to determine if more than two next index accesses have been processed (Figure 4b). If less than two next index accesses have been processed, the output code 204-6 calls the Record File Manager (RFM) 206 to retrieve database keys of a record from an index key value. The RFM 206 in turn calls the I/O Random Controller

208, which is coupled to a Buffer Manager 210 and Buffer pool 212. On the second Search Next request, if certain conditions are met, the output code 204-6 will invoke the subroutine to bypass one or more lower component layers, i.e., the RFM 206, Random Controller 208, the Buffer Manager 210, to retrieve database keys. Nothing in Edwards teaches or suggests any component or subroutine “issuing a callback” to the calling party, as recited in claims 1 and 8.

In the Final Office Action and in the Advisory Action, the Examiner contends that Edwards “discloses issuing the call to the calling program” at column 5, lines 28-33, column 7, lines 10-18 and column 8, lines 35-39. The first portion of Edwards cited by the Examiner describes the relationship between the SQL Director Component layer 202 and the SQL Adapter 201, which “includes a runtime library that contains runtime routines bound into the application used by an application such as a COBOL-85 program for issuing calls. Each such call results in library sending a query statement to the SQL Director component layer 202.” (Col. 5, lines 28-33). The second cited portion describes the Search Next Index function in the RFM layer 206 (Figure 3a). “The Search Next Index function is used to return next key information to the calling program based upon a currency that has been previously established for the index against which this call is issued.” (Col. 7, lines 10-18). The third cited portion describes fine level index entries within a fine level page. “The index entries are lightly packed, adjacent to one another in sorted order as indicated. Thus, if one knows the offset to a particular fine level index entry (i.e., the reference entry), its adjacent entry is found by incrementing the pointer (Col. 8, lines 35-40). None of these cited portions teaches or suggests a component that is called by another component, e.g., library or Search Next Index function, “issuing a callback” to the calling component, as recited in claims 1 and 8.

In the interview, the Examiner stated that Edwards’ Search Next Index function

performed a “callback” when it returned the next key information. The Examiner took the position that a “callback” was equivalent to a response, any response, from a program to the calling program. Applicants respectfully disagree and respectfully submit that a “callback” is a well known term used by those skilled in the art of computer programming and refers to a specific type of response from a called program to the calling program. Specifically, a “callback” is a request from the called program to the calling program asking the calling program to perform some task. Thus, in a callback situation, the originally called program becomes a calling program, and the original calling program becomes a called program.

In the present invention, the data manager (calling program) initially calls the index manager (called program) to locate data. After the index manager locates the data, the index manager calls the data manager, i.e., issues a callback, to determine whether the data will be returned to the runtime. (Specification, page 10, lines 2-11, for example). Thus, the index manager, which is the originally called program, becomes a calling program, and the data manager, which is the original calling program, becomes a called program. Nothing in Ponnekanti or Edwards teaches or suggests such a “callback” between components.

Moreover, Ponnekanti in view of Edwards fails to teach or suggest “continuing to carry out an index-data fetch for another data identifier if . . . the index manager receives a specific condition from the data manager in response to the callback,” as recited in claims 1 and 8. In Ponnekanti, once a RID (e.g., 10.4) for a key value (e.g., 14) is returned to the Data Layer, the Data Layer calls the Index Manager to check if the page time-stamp of the index page has changed. If the time-stamp has changed, the Data Layer then asks the Index Manager to restart the index scan. If the key value/RID (14, 10.4) pair is not found, the Index Manager “continues the scan at the next row.” (Col. 16, lines 12-44). Accordingly, the Index Manager continues to

carry out the index scan only if the Index Manager is unable to locate the key value/RID. The specific condition triggering the continued index scan is from the Index Manager itself. In contrast to the present invention, Ponnekanti's Index Manager does not continue "to carry out an index-data fetch for another data identifier if . . . the index manager receives a specific condition from the data manager in response to the callback," as recited in claims 1 and 8.

Applicants respectfully submit that Ponnekanti in view of Edwards fails to teach or suggest the combination of elements recited in claims 1 and 8. Accordingly, claims 1 and 8 are allowable over the cited references. Claims 2-7, 9-14, 56 and 57 depend on claims 1 and 8, respectively, and the above arguments apply with full force. Therefore, claims 2-7, 9-14, 56 and 57 are also allowable over the cited reference.

C. Dependent Claims 6 and 13 are allowable over the cited references.

Applicants respectfully submit that dependent claims 6 and 13 are allowable over the cited references for reasons in addition to those presented above relating to claims 1 and 8.

Claim 6 provides:

6. The method of claim 2 wherein the index comprises a plurality of pages having index nodes and the method further comprises the steps of:

c) allowing the index manager to stabilize a page containing a node to be accessed in the index to locate the data identifier for the selected key value wherein the index manager does not release the stabilization of the page during a callback to the data manager.

Claim 13 is a computer readable medium claim having similar scope.

In the present invention, during an index scan, the index manager stabilizes, i.e., latches, an index page that contains nodes of the index (the index is a tree structure). When the index manager locates a matching key on a page, it maintains the latch on the data page while it issues

the callback to the data manager. In this way, efficiencies are introduced to query processing. (Specification, page 10, lines 15-20).

Applicants respectfully submit that Ponnetanki and Edwards fails to teach or suggest an index manager that “does not release the stabilization of the [index] page during a callback to the data manager,” as recited in claims 6 and 13. In Ponnetanki, the Index Manager “unlatches the index page before returning the RID to the Data Layer.” (Col. 16, lines 6-7). Edwards makes no mention of latching the index page.

In the Final Office Action, the Examiner states:

With respect to claims 6-7, Ponnekanti discloses a method for processing a database query as discussed in claim 1. Also Ponnekanti discloses B-tree for data page table (col. 9, lines 30-40) and data identifier such as RID: col. 10, lines 30-51).

As to the limitation, “a callback to the data manager,” Ponnekanti does not explicitly indicate the callback.

However, Edwards discloses issuing the call to the calling program for index key value in the searching index key (col. 7, lines 10-18 and col. 5, lines 28-33).

Applicants respectfully agree that the cited portions of Ponnekanti disclose a B-tree structure and data identifiers. Nevertheless, Applicants respectfully submit that the cited portions do not teach or suggest an “index manager [that] does not release the stabilization of the page during a callback to the data manager,” as recited in claims 6 and 13. Accordingly, claims 6 and 13 are allowable over the cited references.

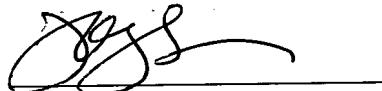
D. Summary of Arguments

For the reasons set forth above, Appellants respectfully submit that the claims 1-14, 56, and 57 are allowable over the cited references. Appellants respectfully request that the final rejection of claims 1-14, 56 and 57 be reversed.

Note: For convenience of detachment without disturbing the integrity of the remainder of pages of this Appeal Brief, Appellants' APPENDIX A is attached on separate sheets following the signatory portion of this Appeal Brief.

This Brief is being submitted in triplicate, and check no. 5790 in the amount of \$330.00 is enclosed for payment of the required Brief fee. Please charge any fee that may be necessary for the continued pendency of this application to Deposit Account No. 02-2120.

Respectfully submitted,



Joyce Tom
Attorney for Applicants
Reg. No. 48,681
(650) 493-4540

IX. APPENDIX A

1. (amended): A method for processing a database query on a set of data in a database management system having a data manager and an index manager, the method comprising the steps of:

- a) in response to a data manager call to locate a data identifier in an index corresponding to a selected key value, performing the steps of:
 - i) locating the data identifier in the index for the selected key value; and
 - ii) issuing a callback to the data manager; and
- b) continuing to carry out an index-data fetch for another data identifier if there is another data identifier for the selected key value in the index and the index manager receives a specific condition from the data manager in response to the callback.

2. (amended): The method of claim 1 wherein step a) further comprises:

- iii) determining whether the data specified by the data identifier is to be returned to a runtime; and
- iv) allowing the data manager to locate the data specified by the identifier and make a determination.

3. (original): The method of claim 2 wherein the determination involves determining whether a predicate check or a data consuming operation on the data are possible.

4. (original): The method of claim 3 wherein the predicate check or the data consuming operation is performed if it is possible to perform the predicate check or the data consuming

operation and a specific condition is returned to the index manager.

5. (original): The method of claim 4 wherein the specific condition comprises a no data return condition.

6. (original): The method of claim 2 wherein the index comprises a plurality of pages having index nodes and the method further comprises the steps of:

c) allowing the index manager to stabilize a page containing a node to be accessed in the index to locate the data identifier for the selected key value wherein the index manager does not release the stabilization of the page during a callback to the data manager.

7. (original): The method of claim 1 wherein the index comprises a plurality of pages having index nodes and the database management system supports a set of access processes potentially able to access the index, the method further comprising the steps of:

c) determining if one of the set of access processes is waiting for an exclusive latch for accessing the index;

d) utilizing the index manager to copy the data identifier and key value from the index if there is such an access process; and

e) releasing stabilization of any pages in the index prior to any callback to the data manager.

8. (amended): A computer readable medium containing program instructions processing a database query on a set of data in a database management system having a data manager and an index manager, the program instructions comprising the steps of:

- a) in response to a data manager call to locate a data identifier in an index corresponding to a selected key value, performing the step of:
 - i) locating the data identifier in the index for the selected key value; and
 - ii) issuing a callback to the data manager; and
- b) continuing to carry out an index-data fetch for another data identifier if there is another data identifier for the selected key value in the index and the index manager receives a specific condition from the data manager in response to the callback.

9. (amended): The computer readable medium of claim 8 wherein step a) further comprises:

- iii) determining whether the data specified by the data identifier is to be returned to a runtime; and
- iv) allowing the data manager to locate the data specified by the identifier and make a determination.

10. (original): The computer readable medium of claim 9 wherein the determination involves determining whether a predicate check or a data consuming operation on the data are possible.

11. (original): The computer readable medium of claim 10 wherein the predicate check or the data consuming operation is performed if it is possible to perform the predicate check or the data consuming operation and a specific condition is returned to the index manager.

12. (amended): The computer readable medium of claim 11 wherein the specific condition comprises a no data return condition.

13. (original): The computer readable medium of claim 9 wherein the index comprises a plurality of pages having index nodes and the program instructions further comprises the steps of:

c) allowing the index manager to stabilize a page containing a node to be accessed in the index to locate the data identifier for the selected key value wherein the index manager does not release the stabilization of the page during a callback to the data manager.

14. (original): The computer readable medium of claim 9 wherein the index comprises a plurality of pages having index nodes and the database management system supports a set of access processes potentially able to access the index, the program instructions further comprising the step of:

c) determining if one of the set of access processes is waiting for an exclusive latch for accessing the index;

d) utilizing the index manager to copy the data identifier and key value from the index if there is such an access process; and

e) releasing stabilization of any pages in the index prior to any callback to the data manager.

56. (new): The method of claim 1, wherein step a) further comprises:

iii) specifying that a callback is required when the data identifier is located.

57. (new): The computer readable medium of claim 8, wherein step a) further comprises:

- iii) specifying that a callback is required when the data identifier is located.